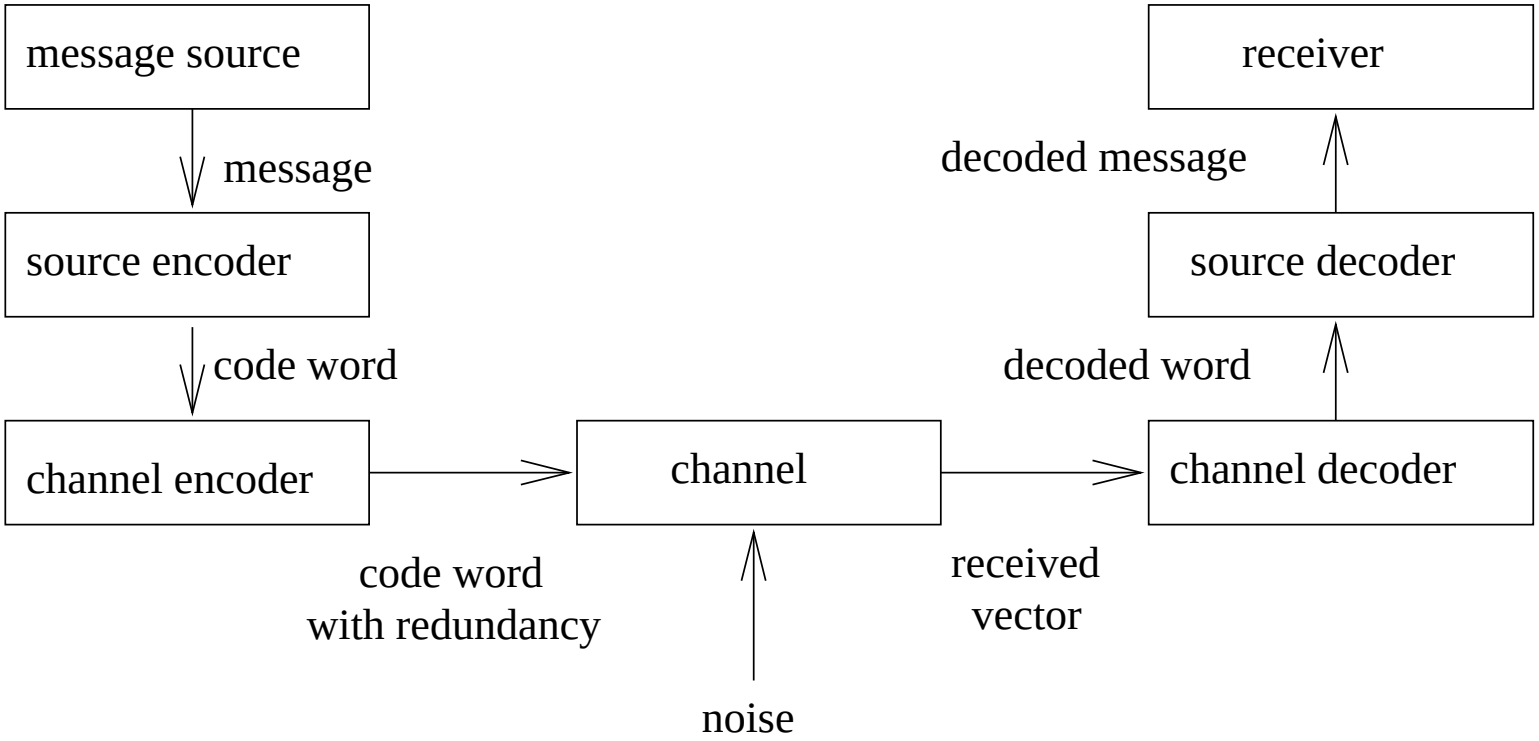



**Figure 1** *Encoding and decoding of message. The channel encoder and decoder are there to introduce redundancy which let us detect and correct errors.*

Criteria for designing channel encoding algorithm and for the construction of the encoder and the decoder are namely fast encoding and decoding of messages, easy transmission of encoded messages, maximum rate of transfer of information, and maximum detection or correction capability.

**Definition 1.** Let $A = \{a_1, \ldots, a_q\}$ be a *code alphabet* of size $q$, and its elements are the *code symbols*. We call a *q-ary word* of length $n$ over $A$ a sequence $\mathbf{w} = w_1 \cdots w_n$, or equivalently a vector $(w_1, \ldots, w_n)$, where $w_i \in A$ for all $i$. We call a *q-ary block code* of length $n$ over $A$ a nonempty set $C$ of $q$-ary words, that is *code words*, all of which is of the same length $n$. The number of code words $C$ contains is the *size* $m$ of $C$, consequently $m = |C|$. The *information rate* of the code $C$ is bskp

$$\frac{(\log_q |C|)}{n}$$

We call an $(n, m) - code$ a code of length $n$ and size $m$.

**Example 1.** A code over the code alphabet $\mathbf{F}_2 = \{0, 1\}$ is called a *binary code*, one over $\mathbf{F}_3 = \{0, 1, 2\}$ is called a *ternary code*. The term *quaternary code* refers to a code over either $\mathbf{F}_4 = \{0, 1, 2, 3\}$ or $\mathbf{Z}_4 = \{0, 1, 2, 3\}$.

**Definition 2.** A *communication channel* consists of a finite *channel alphabet* $A = \{a_1, \ldots, a_q\}$ together with a set of *forward channel probabilities* $p_{a_{ij}}$, such that for all $i$

$$\sum_{j=1}^{q} p_{a_{ij}} = 1$$

where $p_{a_{ij}}$ is the conditional probability that $a_j$ is received, given that $a_i$ is sent. If $\mathbf{x}$ is the word received when a word $\mathbf{c}$ was sent, $e$ is the number of places where $\mathbf{x}$ and $\mathbf{c}$ differ, and $n$ the length of each word, then the forward channel probability is

$$p_{\mathbf{cx}} = p^e(1-p)^{n-e}$$

**Definition 3.** Let $\mathbf{c} = c_1 \cdots c_n$ and $\mathbf{x} = x_1 \cdots x_n$ be words of length $n$. Then a communication channel is said to be *memoryless* if

$$p_{\mathbf{cx}} = \prod_{i=1}^{n} p_{c_i x_i}$$

**Definition 4.** A memory less channel with a channel alphabet of size $q$ is called a *$q$-ary symmetric channel* if each symbol transmitted has the same probability $p < \frac{1}{2}$ of being received in error, and whenever a wrong symbol is received, each of the $q-1$ possible errors is equally likely.

If $p > \frac{1}{2}$, the channel is known to be *useless*.

**Example 2.** The *binary symmetric channel* (BSC) is a memoryless channel having a channel alphabet $\{0, 1\}$ and channel probabilities $p_{01} = p_{10} = p$ and $p_{00} = p_{11} = 1 - p$.

This probability of a bit error $p$ in a BSC is called the *cross-over probability* of the BSC.

**Example 3.** When a received word is not among the vocabulary of the code, the most likely word sent is the one whose $p_{\mathbf{c}_i\mathbf{x}_i}$ is maximum over all $i = 1, \ldots, m$.

A rule for finding the most likely code word sent in case of an error is called a *decoding rule*.

**Definition 5.** The *maximum likelihood decoding* is

$$p_{\mathbf{c}_i^x\mathbf{x}} = \max_{\mathbf{c}\in C} p_{\mathbf{cx}}$$

where $\mathbf{x}$ is the word received.

**Algorithm 1** *Decoding algorithm*

**for** all words $\mathbf{x}_i$ received **do**
  **if** $\mathbf{x}_i$ *is not a valid code word* **then**
    $\mathbf{c}_i^x \leftarrow$*the most likely* $\mathbf{c}_i$ *according to the decoding rule*
  **else**
    $\mathbf{c}_i^x \leftarrow \mathbf{x}_i$
  **endif**
**endfor**

**Example 4.** Two kinds of maximum likelihood decoding are, when it happens that there are more than one word that has the same maximum likelihood, the *complete maximum likelihood decoding* chooses one of them arbitrarily, while the *incomplete maximum likelihood decoding* rejects all of them and asks for a retransmission.

**Definition 6.** Let $\mathbf{x} = x_1 \cdots x_n$ and $\mathbf{y} = y_1 \cdots y_n$ be words of length $n$ over an alphabet $A$. Then the *Hamming distance* between $\mathbf{x}$ and $\mathbf{y}$, denoted $\mathrm{d}(\mathbf{x}, \mathbf{y})$, is the number of places where $\mathbf{x}$ and $\mathbf{y}$ are different from each other, and

$$\mathrm{d}(\mathbf{x}, \mathbf{y}) = \mathrm{d}(x_1, y_1) + \ldots + \mathrm{d}(x_n, y_n)$$

where

$$\mathrm{d}(x_i, y_i) = \begin{cases} 0 & \text{if } x_i = y_i \\ 1 & \text{if } x_i \neq y_i \end{cases}$$

**Theorem 1.** The Hamming distance $\mathrm{d}(\mathbf{x}, \mathbf{c}) = i$ corresponds to the forward channel probability

$$p_{\mathbf{cx}} = p^i(1-p)^{n-i}$$

**Proof.** This is obvious from Definition's 2 and 6. □

**Example 5.** From Definition 6 it follows that

$$0 \leq \mathrm{d}(\mathbf{x}, \mathbf{y}) \leq n$$

$\mathrm{d}(\mathbf{x}, \mathbf{y}) = 0$ if and only if $\mathbf{x} = \mathbf{y}$; and

$$\mathrm{d}(\mathbf{x}, \mathbf{y}) = \mathrm{d}(\mathbf{y}, \mathbf{x})$$

**Example 6.** Let $A$ be the roman alphabet.

If $\mathbf{x}$ = 'breed', $\mathbf{y}$ = 'bread', and $\mathbf{z}$ = 'break', then

$\mathrm{d}(\mathbf{x}, \mathbf{y}) = \mathrm{d}(\mathbf{y}, \mathbf{z}) = 1$, and $\mathrm{d}(\mathbf{x}, \mathbf{z}) = 2$.

On the other hand if $A = \{0, 1, 2, 3, 4, 5, 6\}$, $\mathbf{p} = 24601$ and $\mathbf{q} = 54321$, then

$$\mathrm{d}(\mathbf{p}, \mathbf{q}) = 3$$

**Theorem 2.** Let $\mathbf{x}$, $\mathbf{y}$ and $\mathbf{z}$ be words of length $n$ over $A$. Then the triangular inequality for their mutual Hamming distance holds, that is

$$\mathrm{d}(\mathbf{x}, \mathbf{z}) \leq \mathrm{d}(\mathbf{x}, \mathbf{y}) + \mathrm{d}(\mathbf{y}, \mathbf{z})$$

**Proof.** Let $a = \mathrm{d}(\mathbf{x}, \mathbf{z})$, $b = \mathrm{d}(\mathbf{x}, \mathbf{y})$, and $c = \mathrm{d}(\mathbf{y}, \mathbf{z})$. We have $a \geq 0$, $b \geq 0$ and $c \geq 0$.

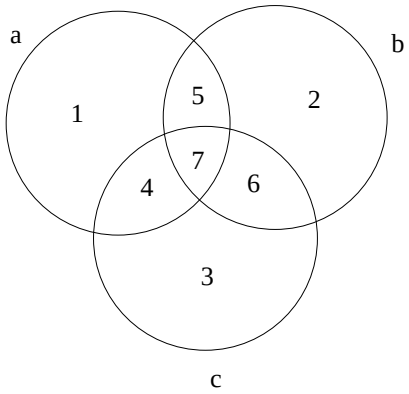
What this theorem states is obvious when $a = 0$.

If $a > 0$, then either $b = 0$ or $b > 0$; if the former is the case, that is $b = 0$, then $a = c$ and the theorem is true.

If both $a > 0$ and $b > 0$, then either $c = 0$ or $c > 0$; if $c = 0$, then $a = b$ and the theorem is again true.

But if $a > 0$, $b > 0$ and $c > 0$, then $a$, $b$ and $c$ may come from some of the diffences in common, as could be shown in the following Venn diagram.

**Figure 2** *Common differences among a, b and c.*



Let $(x, y)$ be the differences in common between distances $x$ and $y$, and similarly $(x, y, z)$ those among $x$, $y$ and $z$.

Then from Figure 2 the area 1 is $(a)$; 2, $(b)$; 3, $(c)$; 4, $(a, c)$; 5, $(a, b)$; 6, $(b, c)$; and 7, $(a, b, c)$.

Then, $\mathrm{d}(\mathbf{x}, \mathbf{z})$ arises from the differences $(a)+(c)+(a, b)+(b, c)$,

$\mathrm{d}(\mathbf{x}, \mathbf{y})$ from $(a) + (b) + (a, c) + (b, c)$,

$\mathrm{d}(\mathbf{y}, \mathbf{z})$ from $(b) + (c) + (a, c) + (a, b)$,

and therefored$(\mathbf{x}, \mathbf{y})+\mathrm{d}(\mathbf{y}, \mathbf{z})$ gives $(a)+(b)+(c)+(a, c)+(a, b)+(b, c)$, which is never less than in the case of $\mathrm{d}(\mathbf{x}, \mathbf{y})$ and hence the theorem is again true. This exhausts all the cases and the theory is proved.

**Definition 7.** The *minimum distance-* or *nearest neighbour* decoding rule decodes $\mathbf{x}$ to $\mathbf{c_x}$ if

$$\mathrm{d}(\mathbf{x}, \mathbf{c_x}) = \min_{\mathbf{c} \in C} \mathrm{d}(\mathbf{x}, \mathbf{c})$$

**Theorem 3.** The maximum likelihood decoding rule and the minimum distance decoding rule is the same for a BSC with cross-over probability

$$p < \frac{1}{2}$$

**Proof.** From Theorem 1, when $p < \frac{1}{2}$, gives

$$p^0(1-p)^n > \cdots > p^n(1-p)^0$$

Thus the less the distance the more the likelihood, and thus the theorem is proved. □

**Definition 8.** Let $C$ be a code containing at least two words. Then, the *minimum distance* or the *distance* of $C$ is

$$\mathrm{d}(C) = \min\{\mathrm{d}(\mathbf{x}, \mathbf{y}) | \mathbf{x}, \mathbf{y} \in C, \mathbf{x} \neq \mathbf{y}\}$$

A code of length $n$, size $m$, and distance $d$ is called

an $(n, m, d)$-*code*.

**Definition 9.** Let a code word be of length $n$.

Then, an *error vector* of weight $k$ is a word containing all the $k$ errors occured taking the value of 1 in their corresponding positions with the remaining positions of the word being zero.

An error vector is also called an *error word* or an *error pattern*.

**Definition 10.** An error vector is said to be *detected* by a code if $a + e$ is not a code word for any code word $a$.

If there exists some code word $a$ such that $a + e$ is also a code word, we say that the error vector $e$ goes *undetected*.

**Definition 11.** Let a received word $\mathbf{x}$ differ from the actual code word sent $\mathbf{c}$ by $e$ errors. Then the corresponding code $C$ is said to be *$u$-error-detecting* if $\mathbf{x}$ is not a code word whenever $1 \leq e \leq u$.

Moreover, $C$ is *exactly $u$-error-detecting* if it is *$u$-error-detecting* but not *$(u+1)$-error-detecting*.

**Theorem 4.** A code $C$ is $u$-error-detecting if and only if

$$\mathrm{d}(C) \geq u + 1$$

**Proof.** Let $\mathbf{c} \in C$.

If $\mathrm{d}(C) \geq u+1$, then $\mathbf{x}$ such that $1 \leq \mathrm{d}(\mathbf{x}, \mathbf{c}) \leq u < \mathrm{d}(C)$ implies that $\mathbf{x} \notin C$, therefore $C$ is $u$-error-detecting.

On the other hand, if $\mathrm{d}(C) < u+1$, that is $\mathrm{d}(C) \leq u$,

then there exist $\mathbf{x}_1, \mathbf{x}_2 \in C$ such that

$1 \leq \mathrm{d}(C) \leq \mathrm{d}(\mathbf{x}_1, \mathbf{x}_2) \leq u$, then it is possible to send $\mathbf{c}_1 \in C$ and incur errors such that

$1 \leq \mathrm{d}(\mathbf{x}, \mathbf{c}_1) = \mathrm{d}(\mathbf{c}_2, \mathbf{c}_1) \leq u$ and $\mathbf{x} = \mathbf{c}_2$, hence $C$ is not a $u$-error-detecting code. ▫

**Corollary 4[1].** A code with distance $d$ is exactly $(d-1)$-error-detecting.

**Definition 12.** Let $v$ be a positive integer and assuming the incomplete decoding rule is used. Then a code $C$ is said to be *$v$-error-correcting* if the minimum distance decoding can correct for it up to $v$ errors. It is said to be *exactly $v$-error-correcting* if it is $v$-error-correcting but not $(v+1)$-error-correcting.

**Theorem 5.** A code $C$ is $v$-error-correcting if and only if

$$\mathrm{d}(C) \geq 2v + 1$$

**Proof.** Suppose that $\mathrm{d}(C) \geq 2v + 1$.

Let $\mathbf{c} \in C$ be the code word sent, $\mathbf{x}$ the word received, and $e$ errors occurred such that $e \leq v$.

Then $\mathrm{d}(\mathbf{x}, \mathbf{c}) \leq v$, and if $C$ is not to be $v$-error-correcting there must be some $\mathbf{c}_1, \mathbf{c}_2 \in C$ such that

$\mathrm{d}(\mathbf{x}, \mathbf{c}_1) + \mathrm{d}(\mathbf{x}, \mathbf{c}_2) \leq 2v$.

But since $\mathrm{d}(C) \geq 2v+1$, which means that $\mathrm{d}(\mathbf{x}, \mathbf{c}_1)+\mathrm{d}(\mathbf{x}, \mathbf{c}_2) \geq 2v + 1$ for all $\mathbf{c}_1, \mathbf{c}_2 \in C$, it follows that $C$ must be $v$-error-correcting.

Next, suppose that $C$ is $v$-error-correcting and

$$\mathrm{d}(C) < 2v + 1$$

Then

$$\mathrm{d}(C) \leq 2v$$

that is to say, there exist $\mathbf{c}_1, \mathbf{c}_2 \in C$ such that $\mathrm{d}(\mathbf{c}_1, \mathbf{c}_2) \leq 2v$. This means that there exist

$\mathbf{x}$ such that $\mathrm{d}(\mathbf{x}, \mathbf{c}_1) + \mathrm{d}(\mathbf{x}, \mathbf{c}_2) = \mathrm{d}(\mathbf{c}_1, \mathbf{c}_2) \leq 2v$, hence $C$ is not $v$-error-correcting. This contradicts what we have supposed earlier, therefore necessarily $\mathrm{d}(C) \geq 2v + 1$. ▫

**Corollary 5[2].** A code with distance $d$ is exactly

$$\left\lfloor \frac{(d-1)}{2} \right\rfloor \text{-error-correcting code,}$$

where $\lfloor x \rfloor$ is the greatest integer less than or equal to $x$.